UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

# Řešení soustav lineárních rovnic s tridiagonální maticí

Jan Holeček

Bakalářská práce
2022

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2021/2022

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

| | |
|---|---|
| Jméno a příjmení: | **Jan Holeček** |
| Osobní číslo: | **I19087** |
| Studijní program: | **B2646 Informační technologie** |
| Studijní obor: | **Informační technologie** |
| Téma práce: | **Řešení soustav lineárních rovnic s tridiagonální maticí** |
| Zadávající katedra: | **Katedra informačních technologií** |

## Zásady pro vypracování

Tridiagonální matice je jedním z typu řídkých matic. Na řešení soustav lineárních rovnic s tridiagonální maticí vedou některé numerické metody např. výpočet splinu, řešení obyčejných diferenciálních rovnic a řešení parciálních diferenciálních rovnic. Soustavy rovnic lze řešit buď přesnými metodami (např. Gaussova eliminace), nebo přibližnými metodami (např. metoda sdružených gradientů). Cílem teoretické části bude porovnání přesných a přibližných metod pro řešení soustav lineárních rovnic s tridiagonální maticí. Cílem aplikační části bude návrh vhodných struktur pro práci a tridiagonální maticí, vytvoření aplikace pro řešení soustav rovnic za využití jednotlivých metod a srovnání jednotlivých metod z hlediska časové a paměťové složitosti.

Rozsah pracovní zprávy: **40 stran**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná**

Seznam doporučené literatury:

YANG, Won-yong. Applied numerical methods using MATLAB. Hoboken, N.J.: Wiley-Interscience, c2005. ISBN 04-716-9833-4. VITÁSEK, Emil. Numerické metody.1. vyd. Praha: SNTL, 1987

Vedoucí bakalářské práce: **RNDr. Josef Rak, Ph.D.**
Katedra matematiky a fyziky

Datum zadání bakalářské práce: **17. prosince 2021**
Termín odevzdání bakalářské práce: **13. května 2022**

L.S.

**Ing. Zdeněk Němec, Ph.D.** v.r.
děkan

**Ing. Jan Panuš, Ph.D.** v.r.
vedoucí katedry

V Pardubicích dne 28. února 2022

Prohlašuji:
Práci s názvem Řešení soustavy lineárních rovnic s tridiagonální maticí jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.
Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 7/2019 Pravidla pro odevzdávání, zveřejňování a formální úpravu závěrečných prací, ve znění pozdějších dodatků, bude práce zveřejněna prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 26. 8. 2022

Jan Holeček

# PODĚKOVÁNÍ

Tímto bych chtěl poděkovat svému vedoucímu bakalářské práce RNDr. Josefu Rakovi, Ph.D. za ochotu a pomoc při vypracování této práce. Děkuji také své rodině za podporu a trpělivost během celého studia.

**ANOTACE**

Bakalářská práce se zaměřuje na řešení soustavy lineárních rovnic pomocí tridiagonální matice. V práci jsou vysvětleny numerické metody pro řešení tridiagonální matice. Dále jsou vysvětleny rozdíly mezi přesnými a přibližnými metodami řešení soustavy rovnic. Pro řešení soustavy rovnic je navržen program v programovacím jazyce Java. Zkoumání vytvořené aplikace pro řešení soustav rovnic za využití jednotlivých metod a srovnání jednotlivých metod z hlediska časové a paměťové složitosti.

**KLÍČOVÁ SLOVA**

Matematické příklady, lineární rovnice, diferenciální rovnice, soustava rovnic, jazyk Java, řešení rovnic, matematické vzorce, tridiagonální matice, Gaussova eliminační metoda, lineární algebra, metoda sdružených gradientů, paměťová složitost, iterační metody, numerické metody, matice, přibližné metody, přesné metody.

**TITLE**

Solution of a system of linear equations using a tridiagonal matrix

**ANNOTATION**

The bachelor thesis focuses on solving a system of linear equations using a tridiagonal matrix. Numerical methods for solving tridiagonal matrices are explained in the work. The differences between the exact and approximate methods of solving the system of equations are also explained. A program in the Java programming language is designed to solve a system of equations. Investigation of the created application for solving systems of equations using individual methods and comparison of individual methods in terms of time and memory complexity.

**KEYWORDS**

Mathematical Examples, Linear Equations, Differential Equations, System of Equations, Java Language, Solving Equations, Mathematical Formulas, Tridiagonal Matrices, Gaussian Elimination Method, Linear Algebra, Conjugate Gradient Method, Memory Complexity, Iterative Methods, Numerical Methods, Matrices, Approximate Methods, Exact methods.

# OBSAH

# SEZNAM OBRÁZKŮ

# SEZNAM TABULEK

# SEZNAM ZKRATEK

| | |
|---|---|
| ODR | Obyčejná diferenciální rovnice |
| GEM | Gaussova eliminační metoda |
| OS | Operační systém |
| PDR | Parciální diferenciální rovnice |
| ODD | Ostře diagonálně dominantní |
| SLR | Soustava lineárních rovnic |
| CGM | Metoda sdružených gradientů |

# ÚVOD

Tato práce má za cíl představit řešení soustavy lineárních rovnic pomocí tridiagonální matice, navrhnout aplikaci pro práci s nimi a následně porovnat jednotlivé metody z hlediska časové a paměťové složitosti.

V teoretické části se zaměříme na soustavu lineárních rovnic, kde si představíme soustavu rovnic a matice. U matic si popíšeme, co to jsou matice, jaké jsou speciální matice a operace s maticemi. Následně si rozebereme přesné metody, u kterých si představíme jednotlivé metody, které do nich patří, jako je Gaussova eliminační metoda a Gauss-Jordanova eliminační metoda. V dalším bodě bude zaměření na přibližné metody, u kterých si opět představíme metody, které do nich patří, jako je Metoda sdružených gradientů, Jacobiho iterace a Gauss-Seidelova iterační metoda.

V následující části si představíme řídké matice, kde si nejprve popíšeme tridiagonální matice a řešení soustavy rovnic pomocí tridiagonální matice. Poté si vysvětlíme parciální diferenciální rovnice, metodu konečných diferencí. Nakonec si popíšeme, co je to spline.

V poslední části teoretické části bude porovnání metod na plně obsazené a řídké matice, kde si nejdříve rozeberme předpoklad pro řešení a následně složitost, jako je časová či výpočetní. Následně si porovnáme paměťovou a časovou složitost jednotlivých metod při řešení řídkých matic.

V aplikační části si představíme programovací jazyk Java, který jsem si vybral pro řešení mé práce. Nejdříve si charakterizujeme programovací jazyk Java, následně si popíšeme Javu platform a nakonec k čemu se Java používá. Následně si popíšeme požadavky, které jsem si stanovil ve své aplikaci. Následně si ukážeme formáty souborů, jaké by měly být, pro vstup do aplikace, aby bylo možné soubory načíst.

# 1 SOUSTAVA LINEÁRNÍCH ROVNIC

## 1.1 Soustava rovnic

Nechť je dána matice A typu m/n, n-členný aritmetický vektor $x = [x_1, x_2, \dots, x_n]^T$ a m-členný aritmetický vektor $b = [b_1, b_2, \dots, b_m]^T$. Potom $Ax = b$ je soustava m lineárních rovnic o n neznámých.

Jestliže $A = [a_{ij}]$, potom $Ax = b$ rozepíšeme do tvaru:

$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1,$

$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2,$

……

$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m,$

Matice $A = [a_{ij}]$ se nazývá matice soustavy.

Vektor $x = [x_1, x_2, \dots, x_n]^T$ se nazývá vektor neznámých.

Vektor $b = [b_1, b_2, \dots, b_m]^T$ se nazývá vektor pravých stran.

Matice $[A|b] = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} & | & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & | & b_2 \\ \vdots & & & \vdots & | & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & | & b_m \end{pmatrix}$ nazýváme rozšířená matice soustavy.

Nechť $Ax = b$ je soustava m lineárních rovnic o n neznámých. Potom každý n-členný aritmetický vektor x, pro který platí $Ax = b$, nazýváme řešení soustavy rovnic. Dvě soustavy lineárních rovnic nazýváme ekvivalentní, jestliže mají stejné množiny řešení.

Jestliže matice $[C|d]$ vznikne z matice $[A|b]$ užitím řádkových elementárních úprav, potom soustavy lineárních rovnic $Ax = b$ a $Cx = d$ jsou ekvivalentní.

Pro sloupcové elementární úpravy podobná věta neplatí!

Soustava se nazývá homogenní, jestliže $b = 0$.

Soustava se nazývá nehomogenní, jestliže $b \neq 0$.

Nechť $Ax = 0$ je homogenní soustava lineárních rovnic, nechť matice soustavy A je typu $m/n$. Potom množina všech řešení homogenní soustavy rovnic je lineární vektorový prostor dimenze $n - h(A)$.

Uvědomme si, že soustava rovnic Ax = b, kde matice soustavy A je typu m/n, je m rovnic pro n neznámých. Proto je dimenze prostoru řešení homogenní soustavy rovnic "počet neznámých – hodnost matice A". Uvědomme si, že homogenní soustavy rovnic mají řešení vždy, jedním řešením je nulový vektor. Jestliže soustava rovnic však bude nehomogenní, pak již řešení mít nemusí.

[25]

## 1.2 Matice

### 1.2.1 Definice

Maticí typu (m, n) nazýváme schéma čísel (reálných nebo i komplexních) sestavených do m řádků n sloupců

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{mn} & a_{m2} & a_{m3} & \cdots & a_{mn} \end{pmatrix}$$

Často používaný zkrácený zápis (pokud víme, o jaký typ matice se jedná) je $A = (a_{ik})$. Je-li $m = n$, pak se A nazývá čtvercovou maticí m-tého stupně (m-tého řádu), jinak hovoříme o matici obdélníkové.
Prvky $a_{11}$, $a_{22}$, $a_{33}$ tvoří hlavní diagonálu a nazývají se hlavní prvky, prvky $a_{1,n}$, $a_{2,n-1}$, $a_{3,n-2}$ tvoří vedlejší diagonálu.

### 1.2.2 Speciální matice

- Nulovou maticí nazýváme matici, jejíž všechny prvky se rovnají nule. Značíme obvykle symbolem 0 nebo i číslem 0.
- Diagonální maticí nazýváme čtvercovou matici, u které prvky na hlavní diagonále jsou různé od nuly a všechny ostatní prvky rovny nule.
- Jednotkovou maticí nazýváme takovou diagonální matici, jejíž všechny hlavní prvky jsou rovny 1. Jednotková matice se obvykle značí E, I nebo i číslem 1.

- Horní (pravou) trojúhelníkovou maticí nazýváme čtvercovou matici $A = (a_{ik})$, jestliže pod hlavní diagonálou jsou všechny prvky nulové, tzn. $\forall i > k: a_{ik} = 0$.
- Dolní (levou) trojúhelníkovou maticí nazýváme čtvercovou matici $A = (a_{ik})$, jestliže nad hlavní diagonálou jsou všechny prvky nulové, tzn. $\forall i < k: a_{ik} = 0$.

[7]

### 1.2.3 Operace s maticemi

#### 1.2.3.1 Sčítání matic

Součtem matic $A = (a_{ij})$ a $B = (b_{ij})$ stejného typu m×n nazýváme
matici $C = A+B$, pro jejíž prvky platí $c_{ij} = a_{ij}+b_{ij}$ $(i = 1, \ldots, m; j = 1, \ldots, n)$.

#### 1.2.3.2 Násobení matice reálným číslem

Součinem reálného čísla $\lambda$ a matice A (nebo také $\lambda$-násobkem matice A) nazýváme matici $C = \lambda \cdot A$, kde $c_{ij} = \lambda \cdot a_{ij}$ $(i = 1, \ldots, n; j = 1, \ldots, m)$.

#### 1.2.3.3 Násobení matic

Předpoklad: $A = (a_{ij})$ je typu $m \times n$, $B = (b_{ij})$ je typu $n \times p$.
Součinem matic A a B pak nazýváme matici $C = A \cdot B$ typu $m \times p$, jejíž prvek $c_{ij}$ je skalárním součinem i-tého řádku z A a j-tého sloupce z B, $(i = 1, \ldots, m; j = 1, \ldots, p)$

### 1.2.4 Pravidla pro operace s maticemi

Předpokládejme, že $\alpha$, $\beta$ jsou reálná čísla a A, B a C jsou matice takové, že níže uvedené operace mají smysl, Pak platí:

a) $A + B = B + A$,
b) $(A + B) + C = A + (B + C)$,
c) $\alpha \cdot (A + B) = \alpha \cdot A + \alpha \cdot B$,
d) $(\alpha + \beta) \cdot A = \alpha \cdot A + \beta \cdot A$,
e) $A \cdot (B + C) = A \cdot B + A \cdot C$,
f) $(A \cdot B) \cdot C = A \cdot (B \cdot C)$,
g) $A \cdot E = A$,
h) $E \cdot B = B$,
i) $(A + B)\, T = AT + BT$,
j) $(A \cdot B)\, T = BT \cdot AT$.

**POZOR!** Násobení matic není komutativní, tj. obecně neplatí, že $A \cdot B = B \cdot A$!

[8]

## 1.3 Přesné metody

### 1.3.1 Gaussova eliminační metoda

Gaussova eliminace je jednou z nejstarších numerických metod, při které se matice soustavy A převádí na horní trojúhelníkovou matici. Řádkovými úpravami s využitím tzv. multiplikátorů se tato matice upraví do tvaru, kdy se pod hlavní diagonálou nachází pouze nuly.
Upravená matice pak odpovídá soustavě rovnic, která je ekvivalentní s původní soustavou, a lze ji řešit podobně jako trojúhelníkovou soustavu lineárních rovnic s pomocí tzv. zpětné substituce (zpětného chodu).

[11]

#### 1.3.1.1 Gaussův algoritmus

**Pivotace**:
Pivotace slouží, aby nevzniklo dělení nulou a tím pádem by nešel udělat 1. krok. Nebo případně k dělení příliš malým číslem, jelikož v případě dělení příliš malým číslem hrozí overflow neboli přetečení a nelze tak vypočítat či určit přesný výsledek.

V původním algoritmu Gaussovy eliminace se hledá nenulový člen v $k$-tem sloupci, aby nedošlo k dělení nulou. Proto je pivotace u tohoto algoritmu nezbytný, protože pivotace „zesiluje“ kontrolu matice a ochranu před chybou.

Nejdříve si určíme P a to tak, že si nejdřív musíme určit $p \in \{k, \dots, n\}$

$$\left|a_{pk}^{(k-1)}\right| = \max_{k \le i \le n} \left|a_{ik}^{(k-1)}\right|$$

Následně prohodíme řádky $k$ a $p$.

1. krok:
Je-li $a_{11} \neq 0$, pak první řádek opíšeme!
Pomocí prvku $a_{11}$ vytvoříme pod ním nuly v prvním sloupci

Je-li $a_{11} = 0$, zaměníme řádky.
Pokud lze, zaměníme tak, aby $a_{11} = 1$ nebo $a_{11} = -1$

2. krok:
Je-li prvek $a'_{22} \neq 0$ (v nové matici),

vytvoříme pomocí něho nuly v druhém sloupci pod ním (druhý řádek opíšeme)
Postup opakujeme, až získáme horní trojúhelníkovou matici.

[11]

**Zpětný chod:**

Pomocí zpětného chodu vypočítáme řešení soustavy.

$$x_n = \frac{y_n}{u_{nn}}$$

Pro $i = n - 1, \dots, 1$

$$x_i = \frac{1}{u_{ii}}\left(y_i - \sum_{j=\mathrm{i}+1}^{n} u_{ij}x_j\right)$$

**Vlastnosti**

Metoda má jedinou podmínku konvergence, a to regulárnost matice A.
Výhodou metody je, že nepotřebuje žádné další místo v paměti, všechny operace se provádí přímo na matici A.
Nevýhodu této metody je asymptotická složitost $O(n^3)$ a nízká numerická stabilita.

[30]

### 1.3.1.2 Příklad

Gaussovou eliminační metodou řešte soustavu

$$x_1 + 2x_2 - 2x_3 = 1,$$
$$x_1 + x_2 + x_3 = 3,$$
$$2x_1 + 2x_2 + x_3 = 5.$$

**Řešení**: Rozšířenou matici soustavy uvedeme na trojúhelníkový tvar. V našem případě řádky postupně násobíme multiplikátory $m^0_{21} = -1$, $m^0_{31} = -2$; $m^1_{32} = -2$. Dostaneme:

$$\begin{pmatrix} 1 & 2 & -2 & | & 1 \\ 1 & 1 & 1 & | & 3 \\ 2 & 2 & 1 & | & 5 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 2 & -2 & | & 1 \\ 0 & -1 & 1 & | & 2 \\ 0 & -2 & 1 & | & 3 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 2 & -2 & | & 1 \\ 0 & -1 & 3 & | & 2 \\ 0 & 0 & -1 & | & -1 \end{pmatrix}$$

Následně pomocí zpětného chodu vypočítáme řešení soustavy rovnic:

$$x_3 = 1,$$
$$x_2 = -2 + 3x_3 = 1,$$
$$x_3 = 1 - 2x_2 + 2x_3 = 1.$$

[8]

### 1.3.2 Gauss-Jordanova eliminační metoda

Gaussovu eliminační metodu lze jednoduše modifikovat způsobem, který je označován jako Gauss-Jordanova metoda. Základní myšlenkou tohoto výpočetního postupu je úprava matice soustavy A na diagonální nebo dokonce jednotkovou matici.

Gauss-Jordanovu metodu je možno použít i k řešení tzv. maticových rovnic, jenž lze zkráceně vyjádřit:

$$A \cdot x = B,$$

Kde X představuje matici kořenů soustavy a B matici levých stran, obě s obecným rozměrem [n, r].

[17]

#### 1.3.2.1 Obecné řešení Gauss-Jordanovy eliminace

Řeším soustavu rovnic A$\vec{x}$ = $\vec{b}$. V prvním kroku, nechť prvek $a_{11} \neq 0$ (lze vždy dosáhnout přehozením rovnic). Prvek $a_{11}$, použitý k úpravě rovnic 2, ..., n nazveme hlavním prvkem (pivot).

Od i-té rovnice odečteme 1. rovnici násobenou multiplikátorem $m_i^{(1)} = -a_{i1}/a_{11}$. Modifikovaná soustava bude mít v 1. sloupci pod diagonálou samé 0. Úprava prováděná současně s pravou stranou odpovídá násobení rovnice maticí

$$D_1 = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ -a_{21}/a_{11} & 1 & 0 & & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ -a_{n1}/a_{11} & 0 & 0 & \cdots & 1 \end{pmatrix}$$

[18]

Výsledkem je diagonální matice. Přičemž lze následně vypočítat jednotkovou matici. Díky tomu zjistíme výsledný vektor, takže není nutný zpětný chod.

$$Ax = \left(\begin{array}{ccccc|c} 1 & 0 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 0 & 8 \\ 0 & 0 & 1 & 0 & 0 & 3 \\ 0 & 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 0 & 1 & 2 \end{array}\right)$$

### 1.3.2.2 Příklad řešení Gauss-Jordanovy eliminace

Máme zadanou soustavu rovnic:

$$3x_1 - 2x_2 + 2x_3 = 10$$
$$x_1 + 3x_2 - x_3 = 2$$
$$2x_1 + 2x_2 + 3x_3 = 15$$

**Postup řešení:**

$$\left(\begin{array}{ccc|c} 3 & -2 & 2 & 10 \\ 1 & 3 & -1 & 2 \\ 2 & 2 & 3 & 15 \end{array}\right) \cdot 3 \sim \left(\begin{array}{ccc|c} 3 & -2 & 2 & 10 \\ 3 & 9 & -3 & 6 \\ 6 & 6 & 9 & 45 \end{array}\right) \begin{array}{l} \\ -r_1 \\ -2r_2 \end{array} \sim \left(\begin{array}{ccc|c} 3 & -2 & 2 & 10 \\ 0 & 11 & -5 & -4 \\ 0 & 10 & 5 & 25 \end{array}\right) \cdot \tfrac{1}{5} \sim \left(\begin{array}{ccc|c} 3 & -2 & 2 & 10 \\ 0 & 11 & -5 & -4 \\ 0 & 2 & 1 & 5 \end{array}\right) \cdot 11 \sim$$

$$\sim \left(\begin{array}{ccc|c} 3 & -2 & 2 & 10 \\ 0 & 11 & -5 & -4 \\ 0 & 22 & 11 & 55 \end{array}\right) - 2r_2 \sim \left(\begin{array}{ccc|c} 3 & -2 & 2 & 10 \\ 0 & 11 & -5 & -4 \\ 0 & 0 & 21 & 63 \end{array}\right) \cdot \tfrac{1}{21} \sim \left(\begin{array}{ccc|c} 3 & -2 & 2 & 10 \\ 0 & 11 & -5 & -4 \\ 0 & 0 & 1 & 3 \end{array}\right) \begin{array}{l} -2r_3 \\ +5r_3 \end{array} \sim \left(\begin{array}{ccc|c} 3 & -2 & 0 & 4 \\ 0 & 11 & 0 & 11 \\ 0 & 0 & 1 & 3 \end{array}\right) \cdot \tfrac{1}{11} \sim$$

$$\sim \left(\begin{array}{ccc|c} 3 & -2 & 0 & 4 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 3 \end{array}\right) + 2r_2 \sim \left(\begin{array}{ccc|c} 3 & 0 & 0 & 6 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 3 \end{array}\right) \cdot \tfrac{1}{3} \sim \left(\begin{array}{ccc|c} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 3 \end{array}\right)$$

Obrázek 1: Příklad řešení soustavy lineárních rovnic pomocí Gauss-Jordanovy eliminace.

Řešení je tedy vektor $x = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$

[19]

## 1.4 Numerické metody

Cílem numerických metod je vytvořit efektivní algoritmy pro řešení nejrůznějších matematických problémů. Formulace úloh a způsob jejich řešení je závislý na skutečnosti, že pracujeme s počítačem. Práce na počítači vyžaduje, abychom zadali na vstup konečný počet číselných údajů a postup prostřednictvím kterého po konečném počtu kroků dojdeme k výsledku. Ten je dán opět konečným počtem výstupních číselných údajů. Jinými slovy, naším cílem je pomocí vhodného algoritmu vyřešit numerickou úlohu.

Numerická úloha je jednoznačný funkční popis vztahu mezi konečným počtem vstupních a konečným počtem výstupních dat.

Jednotlivé matematické úlohy však ještě nemusejí být úlohami numerickými.

Algoritmus je jednoznačně zadaná konečná posloupnost operací, která m-tici přípustných vstupních dat přiřadí n-tici dat výstupních. Neřeší jen jedinou speciální úlohu, ale poskytuje řešení pro celou skupinu úloh téhož typu.

[11]

## 1.5 Přibližné metody

### 1.5.1 Metoda sdružených gradientů

#### 1.5.1.1 Popis

Metoda sdružených gradientů (anglicky conjugate gradient method) se řadí mezi významné iterační metody pro řešení SLR tvaru Ax = b se symetrickou a pozitivně definitní maticí soustavy A. Ve svém článku z roku 1952 metodu poprvé představili Magnus R. Hestenes a Eduard Stiefel. Tato metoda má rychlý algoritmus a lze se přesného výsledku dopočítat již po několika krocích. U matice $n$ x $n$ lze docílit výsledku již po $n$ krocích.

[22]

#### 1.5.1.2 Odvození metody

Koncept metody sdružených gradientů byl původně vyvinut v úzké souvislosti s řešením minimalizačních problémů z oblasti nelineární optimalizace. V dalších odstavcích bude nejprve vysvětlena souvislost mezi úlohou na řešení SLR a problémem hledání lokálních extrémů kvadratické formy. Následně bude na principu tzv. metody největšího spádu představen iterační proces, který spočívá v konstrukci posloupnosti přibližných řešení $\{x_k\}_{k\geq 1}$ dané úlohy. V návaznosti na popis zásadních nevýhod metody největšího spádu bude konečně uveden algoritmus metody sdružených gradientů.

Úloha minimalizace kvadaratické formy.

Nechť $f(x) : R^n \rightarrow R$ je kvadratická funkce $n$ reálných proměnných tvaru

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c, \tag{4.13}$$

kde matice A je symetrická a pozitivně definitní čtvercová matice řádu $n$, symbol b označuje vektor délky $n$ a $c \in R$ je reálný skalár. Zabýváme-li se úlohou hledání extrémů reálné kvadratické formy $f(x)$, pak pro gradient $\nabla f(x)$ z předpisu (4.13) dostáváme

$$\nabla f(x) = \frac{1}{2}A^T x + \frac{1}{2}Ax - b. \tag{4.14}$$

Matice A je navíc symetrická, z čehož plyne

$$\nabla f(x) = Ax - b. \tag{4.15}$$

a z nutné podmínky existence lokálního extrému pak dostáváme

$$\nabla f(x) = Ax - b = 0, \tag{4.16}$$

kde 0 je nulový vektor délky $n$. Z uvedeného postupu tedy plyne, že přesné řešení x * dané SLR tvaru Ax = b je rovněž stacionárním bodem kvadratické formy (4.13). Díky pozitivní definitnosti matice A lze však navíc ukázat, že x * je jediným stacionárním bodem dané kvadratické formy a určuje její globální minimum.

[23]

Konvergence závisí na rozložení vlastních čísel.

$$\lambda_j = \lambda_1 + \frac{j-1}{n-1}(\lambda_n - \lambda_1)\rho^{n-j} \qquad j = 2,3,\dots,n-1.$$

[29]

### 1.5.2 Jacobiho iterace

V případě obecné soustavy lineárních rovnic A · x = b jsou všechny rovnice obecně vyjádřeny:

$a_{i,1} \cdot x_1 + a_{i,2} \cdot x_2 + \cdots + a_{i,n} \cdot x_n = b_i \, , \, i = 1, 2, \ldots, n \, .$ (1.40)

Pokud je $a_{i,i} \neq 0$, lze každou z rovnic upravit:

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j}{a_{i,i}}, \quad i = 1, 2, \ldots, n \, ,$$

což znamená, že i-tý neznámý kořen soustavy.

Jacobiho iterační rekurentní formule pak má tvar:

$$x_i^{(k)} = \frac{b_i - \sum_{j=1}^{i-1} a_{i,j} \cdot x_j^{(k-1)} - \sum_{j=i+1}^{n} a_{i,j} \cdot x_j^{(k-1)}}{a_{i,i}}, \quad i = 1, 2, \ldots,$$

kde k je číslo iteračního cyklu (k = 1, 2, ..., m).

[17]

matice $U_J$ a vektor $v_J$

$$U_J = \begin{pmatrix} 0 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} & \cdots & \frac{a_{1n}}{a_{11}} \\ \frac{a_{21}}{a_{22}} & 0 & \frac{a_{23}}{a_{22}} & \cdots & \frac{a_{n2}}{a_{22}} \\ \frac{a_{31}}{a_{33}} & \frac{a_{32}}{a_{33}} & 0 & \cdots & \frac{a_{3n}}{a_{33}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{a_{n1}}{a_{nn}} & \frac{a_{n2}}{a_{nn}} & \frac{a_{n3}}{a_{nn}} & \cdots & 0 \end{pmatrix}, \; v_J = \begin{pmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{22}} \\ \frac{b_3}{a_{33}} \\ \vdots \\ \frac{b_n}{a_{nn}} \end{pmatrix}$$

kde $a_{ij}$ jsou prvky matice soustavy A

$b_i$ jsou prvky vektoru pravé strany b.

Matici můžeme rozepsat do následujícího tvaru:

$$A = D - L - U,$$

kde D je matice diagonální, L je dolní trojúhelníková matice a U je horní trojúhelníková matice.

$$U = \begin{pmatrix} 0 & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & 0 & a_{23} & \cdots & a_{2n} \\ 0 & 0 & 0 & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}, D = \begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{pmatrix},$$

$$L = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ a_{21} & 0 & 0 & \cdots & 0 \\ a_{31} & a_{32} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & 0 \end{pmatrix}$$

[20]

$$Ax = b \quad \Rightarrow \quad (D - L - U)x = b$$

Je potřeba tuto matici transformovat.

$$Dx = (L + U)x + b$$

Vycházíme z předpokladu, že jednotlivé prvky $a_{ii} \neq 0$, pro i = 1, …, n, je matice D regulární a lze tudíž vypočítat

$$x = D - 1\,(L + U)x + D - 1b$$

a z tohoto vztahu získáme maticový tvar Jacobiho iterační metody.

$$xk + 1 = TJxk + D - 1b, \qquad TJ = D - 1(L + U)$$

Lze si také touto metodou zapsat vektor $\vec{x}$ následovně.

$$\left|x_i^{k+1}\right| = - \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{a_{ij}}{a_{ii}} x_j^k + \frac{b_i}{a_{ii}}, \quad \begin{matrix} i = 1, \ldots, n. \\ k \geq 0 \end{matrix}$$

Pro některé speciální matice A je zaručena konvergence Jacobiho iterační metody.

a) Silné řádkové kritérium:

   Nechť matice A je ryze řádkově diagonálně dominantní, tj.

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, \quad i = 1, \dots, n.$$

   Pak Jacobiho iterační metoda konverguje pro každou počáteční aproximaci $x^0 \in R^n$.

b) Silné sloupcové simulační kritérium:

   Nechť matice A je ryze sloupcově diagonálně dominantní, tj.

$$|a_{kk}| > \sum_{\substack{i=1 \\ i \neq i}}^{n} |a_{ik}|, \qquad k = 1, \dots, n.$$

   Pak Jacobiho iterační metoda konverguje pro každou počáteční aproximaci $x^0 \in R^n$.

[21]

Konvergenci Jacobiovy iterační metody při libovolném počátečním vektoru $x^{(0)}$ pak zaručuje podmínka:

$$\rho(D^{-1}A') < 1$$

1) Postačující podmínka platí ⇒ metoda konverguje
   - matice A je ostře diagonálně dominantní (ODD)
   - existuje (alespoň jedna) norma matice $U_J$ : $||U_J|| < 1$
2) nutná a postačující podmínka platí ⇔ metoda konverguje
   - spektrální poloměr matice $U_J$ : $\rho(U_J) < 1$

Určení spektrálního poloměru matice $U_J$ z matice A.

Vlastní čísla matice $U_J$ učíme z rovnice:

$$\det\begin{pmatrix} \lambda a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & \lambda a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & \lambda a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & \lambda a_{nn} \end{pmatrix} = 0$$

#### 1.5.2.1 Praktický výpočet

Z první rovnice vyjádříme 1. neznámou $x_1^{(k+1)} = \frac{1}{a_{11}}(b1 - a12\,x_2^k - a13\,x_3^k - \dots - a1n\,x_n^k)$

Z druhé rovnice vyjádříme 2. neznámou $x_2^{(k+1)} = \frac{1}{a_{22}}(b2 - a21\,x_1^k - a23\,x_3^k - \dots - a1n\,x_n^k)$

Ze třetí rovnice vyjádříme 3. neznámou $x_3^{(k+1)} = \frac{1}{a_{33}}(b3 - a31\,x_1^k - a32\,x_2^k - \dots - a1n\,x_n^k)$

A tak pokračujeme dále

Postupně (v libovolném pořadí) vypočítáme nové, přesnější, hodnoty $x^{(k+1)}$. Do výrazů na pravé straně dosazujeme za $x^{(k)}$ hodnoty vypočítané na předcházejícím kroku.

[20]

#### 1.5.2.2 Příklad

Máme soustavu lineárních rovnic:

$$11x_1 + 2x_2 + x_3 = 15$$
$$x_1 + 10x_2 + 2x_3 = 16$$
$$2x_1 + 3x_2 - 8x_3 = 1$$

Nyní rovnice převedeme na iterační tvar, tak že z každé rovnice vyjádříme jednu neznámou

$$x_1 = \frac{1}{11}(15 - 2x_2 - x_3)$$
$$x_2 = \frac{1}{10}(16 - x_1 - 2x_3)$$
$$x_3 = \frac{1}{8}(-1 + 2x_1 + 2x_2)$$

$$\begin{pmatrix} x_1^{k+1} \\ x_2^{k+1} \\ x_3^{k+1} \end{pmatrix} = \begin{pmatrix} 0 & -\frac{2}{11} & -\frac{1}{11} \\ -\frac{1}{10} & 0 & -\frac{1}{5} \\ \frac{1}{4} & \frac{3}{8} & 0 \end{pmatrix} \begin{pmatrix} x^4 \\ x_2^k \\ x_3^k \end{pmatrix} + \begin{pmatrix} \frac{15}{11} \\ \frac{16}{10} \\ -\frac{1}{8} \end{pmatrix}$$

$$x_1^{k+1} = \frac{1}{11}\left(15 - x_2^k - x_3^k\right)$$

$$x_2^{k+1} = \frac{1}{10}\left(16 - x_1^k - 2x_3^k\right)$$

$$x_3^{k+1} = \frac{1}{8}\left(- + x_1^k + 2x_2^k\right)$$

Pokud si zvolíme počáteční stav $\vec{x}(0,0,0)$ a hodnotu konečného kritéria $10^4$, potom výsledek po průběhu deseti iterací je $\vec{x}(1.0564, 1.3642, 0.6507)$.

| | $x_1^{(k)}$ | $x_2^{(k)}$ | $x_3^{(k)}$ | $\| x^{(k)} - x^{(k-1)} \|$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | _____ |
| 1 | 1,3636 | 1,6 | -0,125 | 1,60000 |
| 2 | 1,0841 | 1,4886 | 0,8459 | 0,94090 |
| 3 | 1,0188 | 1,3284 | 0,7043 | 0,16020 |
| 4 | 1,0581 | 1,3573 | 0,6279 | 0,07640 |
| 5 | 1,0598 | 1,3686 | 0,6485 | 0,02060 |
| 6 | 1,0558 | 1,3643 | 0,6532 | 0,00470 |
| 7 | 1,0562 | 1,3638 | 0,6506 | 0,00260 |
| 8 | 1,0565 | 1,3643 | 0,6505 | 0,00050 |
| 9 | 1,0565 | 1,3643 | 0,6507 | 0,00020 |
| 10 | 1,0564 | 1,3642 | 0,6507 | 0,00010 |

Tabulka 1: Iterační výpočet SLR pomocí Jacobiovy iterační metody

[21]

Přesné řešení soustavy rovnic je: $x = \begin{pmatrix} 1{,}056443024494143 \\ 1{,}364217252396166 \\ 0{,}650692225772098 \end{pmatrix}$

### 1.5.3 Gauss-Seidelova iterační metoda

Obecnou soustavu lineárních rovnic A · x = b lze opět vyjádřit vztahem (1.40), přičemž lze každou z rovnic definovat při splnění podmínky $a_{i,i} \neq 0$ jako:

$$\boldsymbol{x_i} = \frac{\boldsymbol{b_i} - \sum_{\boldsymbol{j=1}}^{\boldsymbol{i-1}} \boldsymbol{a_{i,j}} \cdot \boldsymbol{x_j} - \sum_{\boldsymbol{j=i+1}}^{\boldsymbol{n}} \boldsymbol{a_{i,j}} \cdot \boldsymbol{x_j}}{\boldsymbol{a_{i,i}}}, \quad i = 1, 2, \ldots,$$

Z této rovnice lze odvodit Gauss-Seidelovu iterační rekurentní formuli:

$$\boldsymbol{x_i^{(k)}} = \frac{\boldsymbol{b_i} - \sum_{\boldsymbol{j=1}}^{\boldsymbol{i-1}} \boldsymbol{a_{i,j}} \cdot \boldsymbol{x_j^{(k)}} - \sum_{\boldsymbol{j=i+1}}^{\boldsymbol{n}} \boldsymbol{a_{i,j}} \cdot \boldsymbol{x_j^{(k-1)}}}{\boldsymbol{a_{i,i}}}, \quad i = 1, 2, \ldots,$$

která se oproti Jacobiho iterační formule odlišuje ve skutečnosti, že vypočtené kořeny soustavy $\boldsymbol{x_j^{(k)}}$ jsou k dalšímu výpočtu použity ihned, a nikoliv až v dalším iteračním cyklu. Celý výpočet pak konverguje k přesnému řešení rychleji.

[17]

Maticový zápis lze zapsat:

$$Ax = b \Rightarrow (D - L - U)x = b$$
$$(D - L)x = Ux + b$$

Za předpokladu, že $a_{ii} \neq 0$, i = 1, …, n, je matice D – L regulární.

$$x = (D - L)^{-1}Ux + (D - L)^{-1}b$$

Potom je Gaussova-Seidelova iterační metoda tvaru.

$$x^{k+1} = T_G x^k + g, \qquad T_G = (D^{-L})^{-1}U, \qquad g = (D - L)^{-1}b$$

Posloupnost $\{x^k\}_{k=0}^{\infty}$ generovaná Gaussovou-Seidelovou iterační metodou konverguje pro každou počáteční aproximaci $x^0 \in R^n$ právě tehdy, když $\rho(T_G) < 1$.

Metodu provedeme na stejném příkladě jako Jacobiovu, opět s počátečními podmínkami $\vec{x}(0,0,0)$ a hodnotu konečného kritéria $10^4$.

**Podmínky konvergence:**

Pro konvergenci Gauss-Seidelovy metody stačí, když platí libovolná z následujících podmínek:

1. A je diagonálně dominantní matice.
2. A je symetrická pozitivně definitní matice.

**Příklad:**

$$11x_1 + 2x_2 + x_3 = 15$$
$$x_1 + 10x_2 + 2x_3 = 16$$
$$2x_1 + 3x_2 - 8x_3 = 1$$

$$x_1^{k+1} = \frac{1}{11}(15 - 2x_2^k - x_3^k)$$

$$x_2^{k+1} = \frac{1}{10}(16 - x_1^{k+1} - 2x_3^k)$$

$$x_3^{k+1} = \frac{1}{8}(-1 + 2x_1^{k+1} + 2x_2^{k+1})$$

| | $x_1^{(k)}$ | $x_2^{(k)}$ | $x_3^{(k)}$ | $\| x^{(k)} - x^{(k-1)} \|$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | _____ |
| 1 | 1,3636 | 1,4636 | 0,7648 | 1,46360 |
| 2 | 1,028 | 1,3442 | 0,6361 | 0,33560 |
| 3 | 1,0614 | 1,3666 | 0,6528 | 0,03340 |
| 4 | 1,0558 | 1,2639 | 0,6504 | 0,10270 |
| 5 | 1,0565 | 1,3643 | 0,6507 | 0,10040 |
| 6 | 1,0564 | 1,3642 | 0,6507 | 0,00011 |
| 7 | 1,0564 | 1,3642 | 0,6507 | 0,00001 |

Tabulka 2: Iterační výpočet SLR pomocí Gauss-Seidelovy iterační metody

Při porovnání obou tabulek výsledků dojdeme k závěru, že výsledek z desáté iterace Jacobiovou a výsledek sedmé iterace Gauss-Seidelovou iterační metodou se rovnají. Stačilo nám k tomu o tři iterace méně, díky použití již vypočítaných hodnot.

$$\vec{x}(1.0564, 1.3642, 0.6507).$$

Přesné řešení soustavy rovnic je: $x = \begin{pmatrix} 1{,}056443024494143 \\ 1{,}364217252396166 \\ 0{,}650692225772098 \end{pmatrix}$

[21],[27]

# 2 ŘÍDKÉ MATICE

## 2.1 Tridiagonální matice

Tridiagonální matice je čtvercová matice, jejíž prvky jsou od hlavní úhlopříčky, podúhlopříčky a nadúhlopříčky vzdáleny nule. Jinými slovy, je to pásková matice s horní a dolní šířkou pásma rovnou 1. Má tvar

$$A = \begin{bmatrix} d_1 & e_1 & & & \\ c_2 & d_2 & e_2 & & \\ & c_3 & \ddots & \ddots & \\ & & \ddots & \ddots & e_{n-1} \\ & & & c_n & d_n \end{bmatrix} \in \mathbb{C}^{n \times n}.$$

Důležitým příkladem je matice Tn, která vzniká při diskretizaci Poissonovy parciální diferenciální rovnice standardním pětibodovým operátorem, ilustrovaná pro n = 5 pomocí

$$T_5 = \begin{bmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & -1 & 4 & -1 & \\ & & -1 & 4 & -1 \\ & & & -1 & 4 \end{bmatrix}$$

Je to symetrická pozitivně definitní, diagonálně dominantní, Toeplitzova matice a M-matice.

Tridiagonální matice mají mnoho speciálních vlastností a existují různé algoritmy, které využívají jejich strukturu.

[9]

### 2.1.1 Řešení soustavy rovnic pomocí tridiagonální matice

Soustava rovnic s tridiagonální maticí. V případě s = 1 hovoříme o tridiagonální matici. Algoritmus GEM pro řešení soustavy rovnic s tridiagonální maticí je velmi jednoduchý. Bez výběru hlavních prvků pro soustavu.

$$\begin{pmatrix} a_1 & c_1 & 0 & 0 & \dots & 0 & 0 & 0 \\ b_1 & a_2 & c_2 & 0 & \dots & 0 & 0 & 0 \\ 0 & b_2 & a_3 & c_3 & \dots & 0 & 0 & 0 \\ \vdots & & \ddots & \ddots & \ddots & & & \\ \vdots & & \ddots & \ddots & \ddots & \ddots & & \\ \vdots & & & & \ddots & \ddots & \ddots & \\ 0 & 0 & 0 & 0 & & b_{n-2} & a_{n-1} & c_{n-1} \\ 0 & 0 & 0 & 0 & & 0 & b_{n-1} & a_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ \vdots \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ \vdots \\ \vdots \\ d_{n-1} \\ d_n \end{pmatrix}$$

V přímém chodu počítáme

$b_i \coloneqq b_i/a_i, \quad a_{i+1} \coloneqq a_{i+1} - b_i c_i, \quad d_{i+1} \coloneqq d_{i+1} - b_i d_i, \qquad i = 1, 2, \dots, n-1,$

a ve zpětném chodu určíme

$x_n \coloneqq \frac{d_n}{a_n}$ a dále $\qquad x_i \coloneqq (d_i - c_i x_{i+1})/a_i, \quad i = n-1, n-2, \dots, 1.$

Transformovaná matice soustavy obsahuje koeficienty LU rozkladu původní matice.

[10]

## 2.2 Parciální diferenciální rovnice

Je to rovnice pro neznámou funkci u více než jedné proměnné, která obsahuje alespoň jednu její parciální derivaci. Matematická definice může vypadat například takto.

Buď n ∈ $\mathbb{N}$, $\Omega \subset \mathbb{R}^d$ otevřená množina, d ≥ 2. Parciální diferenciální rovnicí (dále PDR) pro neznámou funkci u: $\Omega \to \mathbb{R}$ nazvu výraz tvaru

$$F(x, u(x), Du(x), \cdots, D(n-1)\, u(x), D(n)\, u(x)) \;=\; 0\,, \tag{1.2}$$

kde

$$F\colon \Omega \times \mathbb{R} \times \mathbb{R}d \times \cdots \times \mathbb{R}d\hat{}(n-1) \times \mathbb{R}d\hat{}n \to \mathbb{R} \tag{1.3}$$

je daná funkce. Řád rovnice (1.2) je roven řádu nejvyšší derivace u, která se vyskytuje v (1.2)

[6]

## 2.3 Metoda konečných diferencí

Uvažujme modelový případ: lineární parciální diferenciální rovnici 2. řádu eliptického typu:

$$\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} + \sigma(x,y)u = f(x,y), \tag{1.4}$$

Kde u = u(x, y) je definována na oblasti Ω = {(x, y) : a ≤ x ≤ b, c ≤ y ≤ d}, a funkce σ(x, y) ≥ 0 ∀x, y $\epsilon$ Ω, σ, $f$ jsou spojitá na Ω.

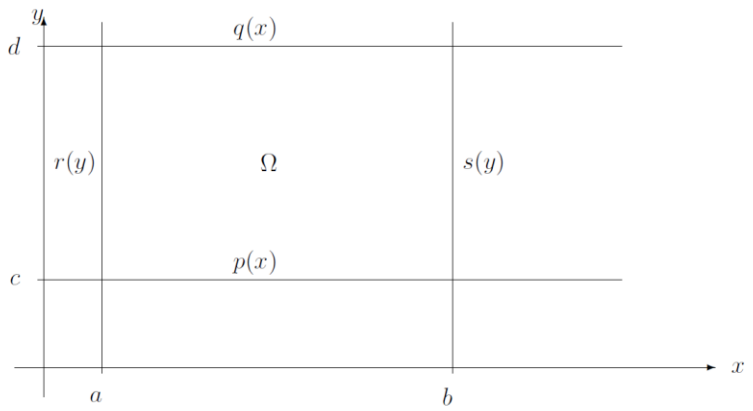
Nechť je splněna tzv. Dirichletova okrajová podmínka na hranicích oblasti Ω:

$$u(x,c) = p(x), \qquad u(x,c) = p(x), \qquad a \leq x \leq b,$$

$$u(a,y) = r(y), \qquad u(b,y) = s(y), \qquad c \leq y \leq d,$$

$$p(a) = r(c), \quad p(b) = s(c), \quad q(a) = r(d), \quad q(b) = s(d).$$

Poslední řádek nám zabezpečuje spojitost okrajových podmínek v „rozích“ oblasti Ω. Viz vzorec 1.4.




Obrázek 2: Metoda konečných diferencí

Vytvoříme síť na oblasti Ω (nejčastěji se používají čtvercové anebo obdélníkové sítě).

$$x_i = s + ih, \quad i = 0, 1, \ldots, n, n+1, \quad h = \frac{b-a}{n+1},$$

$$y_j = c + jk, \quad j = 0, 1, \ldots, m, m+1, \quad k = \frac{d-c}{m+1}.$$

Uzly jsou pak body ($x_i$, $y_j$). Označme $u_{ij} = u(x_i, y_j)$. Jestliže předpokládáme spojitost u(x, y), pak pro dostatečně malé h, k můžeme zanedbat chybové funkce. Potom odvozením do (1.4) dostáváme pro i = 1, 2, …, n, j = 1, 2, …, m:

$$\frac{2u_{ij} - u_{i+1,j} - u_{i-1,j}}{h^2} + \frac{2u_{ij} - u_{i,j+1} - u_{i,j-1}}{k^2} + \sigma_{ij}u_{ij} = f_{ij}.$$

Pro pravidelné čtvercové sítě, tj. h = k, se tato soustava dále zjednodušuje na tvar

$$(4 + h^2\sigma_{ij})u_{ij} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1} = h^2 f_{ij}.$$

Všimněte si, že matice soustavy je v obou případech pro $\sigma_{ij} \not\equiv 0$ diagonálně dominantní, a proto můžeme použít i iterační metody řešení.

Analogický postup můžeme použít pro numerické řešení všech lineárních parciálních diferenciálních rovnic druhého řádu. Postup řešení je vždy stejný: derivace nahradíme diferencemi a hledáme řešení soustavy lineárních algebraických rovnic.

Pro další parciální derivace se používají aproximace:

$$\frac{\partial u(x_j, y_j)}{\partial x} = \frac{u_{i+1,j} - u_{ij}}{h}, \quad \frac{\partial u(x_i, y_j)}{\partial y} = \frac{\mathrm{u_{i,j+1} - u_{ij}}}{\mathrm{k}},$$

$$\frac{\partial \mathrm{u(x_j, y_j)}}{\partial \mathrm{x}} = \frac{\mathrm{u_{ij} - u_{i-1,j}}}{\mathrm{h}}, \quad \frac{\partial \mathrm{u(x_i, y_j)}}{\partial \mathrm{y}} = \frac{\mathrm{u_{ij} - u_{i,j-1}}}{\mathrm{k}},$$

$$\frac{\partial \mathrm{u(x_j, y_j)}}{\partial \mathrm{x}} = \frac{\mathrm{u_{i+1,j} - u_{i-1,j}}}{2\mathrm{h}}, \quad \frac{\partial \mathrm{u(x_i, y_j)}}{\partial \mathrm{y}} = \frac{\mathrm{u_{i,j+1} - u_{i,j-1}}}{2\mathrm{k}},$$

$$\frac{\partial^2 \mathrm{u(x_j, y_j)}}{\partial \mathrm{x}\,\partial \mathrm{y}} = \frac{\mathrm{u_{i+1,j+1} - u_{i+1,j-1} - u_{i-1,j+1} + u_{i-1,j-1}}}{2\mathrm{h}2\mathrm{k}}.$$

Problémy při řešení mohou vznikat, pokud oblast Ω není obdélníková.

[27]

## 2.4 Spline

### 2.4.1 Spline křivky

- jsou většinou intuitivně chápány jako "po částech polynomické funkce se spojitou derivací co do nejvyššího řádu"
- jsou matematickým modelem chování pružného laťkového křivítka, které v minulosti používali konstruktéři trupů lodí
- jsou interpolačními křivkami daného stupně, které interpolují zadané opěrné body

pro dané opěrné body $P_0, \ldots, P_n$ a jim přiřazené číselné hodnoty (parametry) $t_0, \ldots, t_n$, má spline parametrické vyjádření

$$P = P(t), t \in \langle t_0, t_n \rangle,$$

pro které platí $P_i = P(t_i), i = 0, \ldots, n$

volba parametrů:

- uniformní parametrizace, kde $t_{i+1} - t_i = \text{konst.}$
- neuniformní, např. chordálová parametrizace, kde $t_{i+1} = t_i + k\,|P_{i+1} - P_i|$

[12]

### 2.4.2 Přirozený spline

Přirozený spline má tu vlastnost, že pro něj platí $s''(a_0) = s''(a_n) = 0$.
Soustava rovnic, kde $б_i$ je konstanta:

$$h_{i-1}б_{i-1} + 2(h_{i-1} + h_i)б_i + h_i б_{i+1} = i - \Delta_{i-1}, \qquad i = 1, \ldots, n-1,$$

s připojenými rovnicemi $б_0 = б_n = 0$, má tridiagonální matici a je symetrická a diagonálně dominantní. Pro Každou rozumnou volbu uzlů je také dobře podmíněná, takže její vyřešení např. Gaussovou eliminační metodou nepředstavuje žádný problém.

[13]

# 3 POROVNÁNÍ METOD NA PLNĚ OBSAZENÉ A ŘÍDKÉ MATICE

## 3.1 Předpoklad

Pro řešení řídkých matic jsou nejideálnější přibližné metody, jako jsou Jacobiho iterace nebo Gauss-Seidelova iterace z důvodu rychlosti při řešení. Gauss-Jordanova iterační metoda se nedoporučuje z důvodu nepřesnosti výsledku.
Pro řešení plných matic je nejideálnější metoda Gauss-Jordanova a Gaussova eliminační metoda.

## 3.2 Složitost

Výpočetní složitosti podle počtu operací či velikosti paměti potřebné pro konkrétní algoritmus. Oba tyto přístupy mají svá velká omezení, a to velikost komunikace, latence paměti, latence meziprocesorové komunikace i možností využití segmentace operací. Stanovení výpočetní složitosti algoritmu je pak obvykle součástí mnohem obecnější procedury, analýzy algoritmu, ve které se zkoumají všechny prostředky, které konkrétní algoritmus na konkrétní počítačové architektuře potřebuje.
Složitost nejhoršího případu je omezení na čas nebo velikost prostoru, které stačí pro vyřešení úlohy s libovolným vstupem algoritmu z množiny jeho přípustných vstupů. V některých situacích je z praktického pohledu zajímavější průměrná složitost algoritmu, to jest průměrný spotřebovaný čas nebo prostor, který můžeme předpokládat, že algoritmus využije při nějakém rozložení vstupů algoritmu.
Přímočaré algoritmy mají někdy vysokou prostorovou složitost, kterou snížíme teprve přechodem ke komplikovanějším algoritmickým postupům. Časová složitost provádění operací je obvykle méně zřejmá, a i z tohoto důvodu, hovoříme-li dále o složitosti algoritmů bez bližšího vymezení, máme na mysli jejich časovou složitost.

[26]

## 3.3 Porovnání paměťové a časové složitosti

Pro práci s maticemi je nejideálnější z hlediska paměťové složitosti, použití buďto **GEM** nebo **Gauss-Jordanovy metody**, a to z důvodu, že u těchto metod není třeba udržovat cokoliv v paměti, protože všechny operace se provádějí přímo na matici **A**.

**Jacobiho iterace:** Metoda může být velmi rychlá, ale také může být velmi pomalá. Záleží, jakou požadujeme přesnost. Potřebuje v paměti udržovat složky vektoru jak $\boldsymbol{x}^{(i)}$, tak i $\boldsymbol{x}^{(i+1)}$, což je sice nevýhoda, ale nijak veliká, jelikož vektory jsou oproti maticím nenáročné na paměť.

**Gauss-Seidelova metoda:** Metoda má téměř stejné vlastnosti jako Jacobiho iterace. Tato metoda má jedinou výhodu, a to menší paměťovou náročnost, stačí si pamatovat pouze jeden vektor $\boldsymbol{x}$.

**CGM:** Algoritmus je velmi rychlý, lze dokázat, že po *n* krocích vede k přesnému řešení (pokud nezaokrouhlujeme). Metoda požaduje uchovávat v paměti dva vektory o velikosti *n*.

U všech metod je asymptotická složitost $\underline{O(n^3)}$.

Shrnutí všech metod je obsaženo v následující tabulce

| **Metoda** | **Paměťová složitost** | **Časová složitost** |
|---|---|---|
| GEM | Není potřeba další místo v paměti | $O(n^3)$ |
| Gauss-Jordanova metoda | Není potřeba další místo v paměti | $O(n^3)$ |
| CGM | Potřeba udržet v paměti dva vektory o velikosti *n* | $O(n^3)$ |
| Jacobiho iterace | Metoda potřebuje v paměti udržovat složky vektoru $\boldsymbol{x}^{(i)}$ a $\boldsymbol{x}^{(i+1)}$. | $O(n^3)$ |
| Gauss-Seidelova metoda | Potřeba udržet v paměti pouze vektor $\boldsymbol{x}$ | $O(n^3)$ |

Tabulka 3: Obecné porovnání paměťové a časové složitosti metod

[30]

**Porovnání paměťové a časové složitosti řídké matice $T_5$ v kapitole 2.1**

| Metoda | Paměťová složitost | Časová složitost |
|---|---|---|
| GEM | 104 bitů | 2 ms |
| Gauss-Jordanova metoda | 104 bitů | 3 ms |
| CGM | 16 bitů | 2 ms |
| Jacobiho iterace | 16 bitů | 2 ms |
| Gauss-Seidelova metoda | 16 bitů | 2 ms |

Tabulka 4: Porovnání paměťové a časové složitosti řídké matice 5x5

**Porovnání paměťové a časové složitosti řídké matice o velikost 50x50**

| Metoda | Paměťová složitost | Časová složitost |
|---|---|---|
| GEM | 1 184 bitů | 67 ms |
| Gauss-Jordanova metoda | 1 184 bitů | 69 ms |
| CGM | 16 bitů | 2 ms |
| Jacobiho iterace | 16 bitů | 535 ms |
| Gauss-Seidelova metoda | 16 bitů | 380 ms |

Tabulka 5: Porovnání paměťové a časové složitosti řídké matice 50x50

**Porovnání paměťové a časové složitosti řídké matice o velikost 100x100**

| Metoda | Paměťová složitost | Časová složitost |
|---|---|---|
| GEM | 2 384bitů | 529 ms |
| Gauss-Jordanova metoda | 2 384 bitů | 670 ms |
| CGM | 16 bitů | 105 ms |
| Jacobiho iterace | 16 bitů | 2 339 ms |
| Gauss-Seidelova metoda | 16 bitů | 1 522 ms |

Tabulka 6: Porovnání paměťové a časové složitosti řídké matice 100x100

# 4 APLIKAČNÍ ČÁST

## 4.1 Volba programovacího jazyka

Vzhledem k mým zkušenostem a znalostem jednotlivých programovacích jazyků jsem přemýšlel mezi dvěma:

- Java
- C#

Jazyk Java má oproti jazyku C# několik nevýhod, jako je například vysoká paměťová náročnost aplikací a nemožnost přetěžování operátorů.

Nakonec jsem si vybral pro svoji aplikaci programovací jazyk Java. Javu jsem vybral výhradně kvůli svým zkušenostem.

## 4.2 Programovací jazyk Java

### 4.2.1 Charakteristika

Java je multiplatformní, objektově orientovaný a síťově orientovaný jazyk. Patří mezi nejpoužívanější programovací jazyky. Java se také používá jako výpočetní platforma.
Je považován za jeden z rychlých, bezpečných a spolehlivých programovacích jazyků, který většina organizací preferuje při sestavování svých projektů.

[1]

Když programátor píše aplikaci Java, zkompilovaný kód (známý jako bytecode) běží na většině operačních systémech (OS), včetně Windows, Linux a Mac OS. Java odvozuje většinu své syntaxe z programovacích jazyků C a C++.

[2]

Aktuálně nejnovější verze Java je Java SE 17 (14. 9. 2021)

[3]

### 4.2.2 Java platform

Java Platform je soubor programů, které pomáhají programátorům efektivně vyvíjet a provozovat programovací aplikace Java. Obsahuje spouštěcí stroj, kompilátor a sadu knihoven. Jedná se o soubor počítačového softwaru a specifikací. James Gosling vyvinul platformu Java ve společnosti Sun Microsystems a později ji získala společnost Oracle.

[1]

### 4.2.3 K čemu se Java používá

- Používá se pro vývoj aplikací pro Android
- Pomáhá vám vytvářet podnikový software
- Široká škála mobilních Java aplikací
- Aplikace pro vědecké výpočty
- Použijte pro analýzu velkých dat
- Java programování hardwarových zařízení
- Používá se pro technologie na straně serveru, jako je Apache, JBoss, GlassFish atd.

[1]

## 4.3 Požadavky

Požadavky, které jsem si vytyčil pro mnou implementovanou aplikaci:

- Aplikace by měla mít přehlednou grafickou verzi
- Vstupy a výstupy by aplikace měla mít pouze v csv souborech

## 4.4 Formáty souborů s daty pro aplikaci

### 4.4.1 Matice

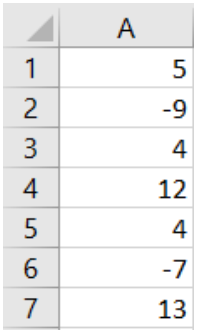
Matice musí začínat v souboru v buňce A1. Pokud bude matice začínat na jiných souřadnicích, tak v aplikaci se objeví chybová hláška, informující uživatele, že matici nelze načíst. Správný formát matice je znázorněn na následujícím obrázku Obrázek 3: Názorná ukázka formátu matice v souboru.

| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 5 | 4 | -8 | 3 | 2 |
| 2 | 9 | 8 | 8 | 5 | 6 | -2 | 4 |
| 3 | 1 | 12 | 6 | 23 | 5 | 8 | -4 |
| 4 | -9 | 8 | -9 | 5 | 43 | 2 | 4 |
| 5 | 3 | 8 | -7 | -9 | -12 | 3 | 7 |
| 6 | 5 | 3 | 8 | -6 | 2 | 9 | 4 |
| 7 | -3 | 9 | -4 | 12 | 2 | 5 | -23 |

Obrázek 3: Názorná ukázka formátu matice v souboru

### 4.4.2 Vektor

Vektor musí stejně jako matice začínat v buňce A1. Také pokud bude vektor začínat na jiných souřadnicích, v aplikaci se zobrazí chybová hláška, která informuje uživatele, že vektor nebyl načten. Správný formát vektoru je znázorněn na následujícím obrázku Obrázek 4: Názorná ukázka formátu vektoru v souboru.



| | A |
|---|---|
| 1 | 5 |
| 2 | -9 |
| 3 | 4 |
| 4 | 12 |
| 5 | 4 |
| 6 | -7 |
| 7 | 13 |

Obrázek 4: Názorná ukázka formátu vektoru v souboru

# ZÁVĚR

Cílem teoretické části této bakalářské práce bylo srovnat některé algoritmy pro řešení soustav lineárních rovnic. Nejprve jsem rozebral soustavu lineárních rovnic. Co jsou to matice a operace s nimi. Následně jsem rozebral přesné metody, do kterých patří Gaussova eliminační metoda či Gauss-Jordanova metoda. Dále jsem rozebral numerické metody a následně přibližné metody, kam patří například CGM, Jacobiho iterace či Gauss-Seidelova iterační metoda. Jako další věc jsem se zaměřil na řídké matice. A jako poslední věc teoretické části jsem porovnal metody na plně obsazené a řídké matice. Následně jsem porovnal jednotlivé metody z hlediska paměťové a časové složitosti.

Pro malé soustavy rovnic je celkem jedno, jakou metodu použijeme, protože rozdíly ve výpočetním čase a nutné paměti budou velice malé. Pro velké řídké matice je nejideálnější využití přibližných metod, protože v případě použití přesné metody, bude výsledek nepřesný. Pro plně obsazené matice se doporučuje použití přesné metody.

Pro řídké matice vychází jako nejlepší metoda jak časově, tak paměťově Metoda sdružených gradientů – viz Tabulka 3: Obecné porovnání paměťové a časové složitosti, Tabulka 4: Porovnání paměťové a časové složitosti řídké matice 5x5, Tabulka 6: Porovnání paměťové a časové složitosti řídké matice 100x100, CGM má ovšem limit, že matice musí být symetrická a pozitivně definitní. U Jacobi a Gauss-Seidelovy iterační metody jde paměťová a časová složitost proti sobě – viz tabulky.

Cílem aplikační části bylo vytvoření aplikace pro řešení soustav lineárních rovnic s tridiagonální maticí. Nejdříve jsem rozebral programovací jazyk Java. Dále jsou popsány správné soubory s daty a jaké typy souborů jsou možné pro použití v aplikaci.

Uživatelská příručka aplikace je uvedena v příloze. Zdrojové kódy aplikace v jazyce Java jsou uloženy v přiloženém ZIP souboru.

# POUŽITÁ LITERATURA


[1] HARTMAN, James. *What is Java?* Definition, Meaning & Features of Java Platforms. Guru99 [online]. 12. 2. 2022 [cit. 2022-03-24]. Dostupné z: https://www.guru99.com/java-platform.html

[2] STOLTZUS, Justin. *Java*. Techopedia [online]. 27. 11. 2020 [cit. 2022-03-24]. Dostupné z: https://www.techopedia.com/definition/3927/java

[3] *Java Downloads*. Oracle [online]. ©2022 [cit. 2022-03-24] Dostupné z: https://www.oracle.com/java/technologies/downloads/

[4] *Lineární rovnice*. E-learning VŠCHT [online]. 12. 8. 2020 [cit. 2022-03-24] Dostupné z: https://e-learning.vscht.cz/mod/page/view.php?id=6190

[5] JAREŠOVÁ, Miroslava a VYBÍRAL, Bohumil. *Diferenciální rovnice.* Fyzikální olympiáda [online]. ©2002–2022 [cit. 2022-03-24]. Dostupné z: http://fyzikalniolympiada.cz/texty/matematika/difro.pdf

[6] ROKYTA, Mirko. *Parciální diferenciální rovnice I*. Univerzita Karlova [online]. 14. 1. 2011 [cit. 2022-03-24]. Dostupné z: https://www2.karlin.mff.cuni.cz/~rokyta/vyuka/1011/zs/Pdr1/texopis/PDR1-20110114.pdf

[7] REHANEK. *Matice – základní definice*. Ostravská univerzita, katedra fyziky [online]. 2005 [cit. 2022-03-24]. Dostupné z: http://artemis.osu.cz/mmmat/txt/la/mzp.htm

[8] MRÁZ, František. *Matice.* České vysoké učení technické v Praze, Fakulta strojní [online]. 2017 [cit. 2022-03-24]. Dostupné z: https://mat.nipax.cz/_media/matice_2018.pdf

[9] HIGHMAN, Nick. *What Is a Tridiagonal Matrix?* Applied mathematics, numerical linear algebra and software [online]. 10. 1. 2022 [cit. 2022-03-24]. Dostupné z: https://nhigham.com/2022/01/10/what-is-a-tridiagonal-matrix/

[10] ČERMÁK, Libor a HLAVIČKA, Rudolf. *Numerické metody*. Ústav matematiky FSI VUT Brno [online]. 6. 2. 2006 [cit. 2022-03-24]. Dostupné z: https://mathonline.fme.vutbr.cz/default.aspx?section=8&server=1&article=11&chapter=119

[11] MOŠOVÁ, Vratislava. *Numerické metody*. Univerzita J. E. Purkyně v Ústí nad Labem, Přírodovědecká fakulta [online]. [cit. 2022-03-24]. Dostupné z: http://physics.ujep.cz/~jskvor/NME/DalsiSkripta/numerickemetody.pdf

[12] *Spline křivky*, Fakulta aplikovaných věd Západočeské univerzity v Plzni, Katedra matematiky [online]. [2019] [cit. 2022-03-24]. Dostupné z: http://home.zcu.cz/~bastl/GM1/GM1_lecture03.pdf

[13] VITÁSEK, Emil. *Numerické metody*. 67. Praha: SNTL – Nakladatelství technické literatury, 1987, 512 s. ISBN 04-009-87.

[14] DOLEŽALOVÁ, Jarmila. *Diferenciální rovnice*. Vysoká škola báňská – Technická univerzita Ostrava, Katedra matematiky a deskriptivní geometrie [online]. [2018] [cit. 2022-03-24]. Dostupné z: https://homel.vsb.cz/~dol30/MII-difrovnice.pdf

[15] ŽELEZNÝ, Zdeněk. *Diferenciální rovnice 1. řádku, sbírka řešených příkladů* [online]. České Budějovice, 2012 [cit. 2022-03-24]. Dostupné z: https://theses.cz/id/kgrf4r/Zdenk_elezn_Diferenciln_rovnice_1__du_-_Sbrka_eench_pklad.pdf. Diplomová práce. Jihočeská univerzita v Českých Budějovicích, Pedagogická fakulta. Vedoucí práce RNDr. Libuše Samková, Ph.D.

[16] SCHREIBEROVÁ, Petra a VOLNÝ, Petr. *Obyčejné diferenciální rovnice*. Vysoká škola báňská – Technická Univerzita Ostrava [online]. 2012 [cit. 2022-03-24]. Dostupné z: https://theses.cz/id/kgrf4r/Zdenk_elezn_Diferenciln_rovnice_1__du_-_Sbrka_eench_pklad.pdf

[17] KREJSA, Martin. *Soustavy lineárních rovnic*. Vysoká škola báňská – Technická univerzita Ostrava, Fakulta stavební [online]. [2018] [cit. 2022-03-25]. Dostupné z: http://fast10.vsb.cz/krejsa/studium/algoritmy_05.pdf

[18] LIMPOUCH, Jiří. *Gaussova a Gauss-Jordanova eliminace*. České vysoké učení technické v Praze, Katedra fyzikální elektroniky [online]. 8. 3. 2000 [cit. 2022-03-25]. Dostupné z: http://pascal.fjfi.cvut.cz/~limpouch/numet/linalg/node7.html

[19] VONDRÁK, Vít. *Gaussova-Jordanova eliminační metoda*. Vysoká škola báňská – Technická Univerzita Ostrava [online]. [cit. 2022-03-25]. Dostupné z: https://homel.vsb.cz/~ber95/LA/Cviceni/lacv4.pdf

[20] MAJLINGOVÁ, Olga. *Iterační metody, Jacobiho a Gaussova-Seidelova*. Fakulta strojní ČVUT v Praze [online]. [cit. 2022-03-25]. Dostupné z: https://olga.majling.eu/Vyuka/NM_20_cv2c.pdf

[21] BAŤKA, Václav. *Iterační metody řešení soustav lineárních rovnic*[online]. Pardubice, 2012 [cit. 2022-03-25]. Dostupné z: https://dk.upce.cz/bitstream/handle/10195/46579/BatkaV_IteracniMetody_JH_2012.pdf?sequence=2&isAllowed=y. Bakalářská práce Univerzita Pardubice, Fakulta elektrotechniky a informatiky. Vedoucí práce Mgr. Jana Heckenbergerová, Ph.D.

[22] HESTENES, Magnus R. a Eduard STIEFEL. *Methods of conjugate gradients for solving linear systems.* Journal of Research of the National Bureau of Stan- dards. 1952, 49(6), 409-435. DOI: http://dx.doi.org/10.6028/jres.049.044.

[23] SHEWCHUK, Jonathan Richard. *An Introduction to the Conjugate Gradi- ent Method Without the Agonizing Pain* [online]. Edition $1\frac{1}{4}$. Technical re- port CMU-CS-94-125. Pittsburgh, PA, USA: Carnegie Mellon University, 1994. [cit. 2022-03-21]. Dostupné z: https://www.cs.cmu.edu/q̃uake-papers/painless- conjugate-gradient.pdf

[24] BRYAN, Kurt. *Conjugate Gradient Methods*. In: Rose-Hulman Institute of Technology [online]. Terre Haute: Department of Mathematics, Rose-Hulman Institute of Technology, 2011 [cit. 2022-03-21]. Dostupné z: https://www.rosehulman.edu/b˜ryan/lottamath/congrad.pdf

[25] TESKOVÁ, Libuše. *Soustavy lineárních rovnic*. Západočeská univerzita v Plzni [online]. [cit. 2022-04-11]. Dostupné z: http://home.zcu.cz/~teskova/WWW-KMA/DALKA3.pdf

[26] TŮMA, Miroslav. *Řídké matice v řešení soustav lineárních algebraických rovnic*. Univerzita Karlova, Matematicko-fyzikální fakulta [online]. 26. 8. 2020 [cit. 2022-05-03]. Dostupné z: https://www2.karlin.mff.cuni.cz/~mirektuma/rm/rmatice.pdf

[27] LIMPOUCH, Jiří. *Gauss-Seidelova metoda*. České vysoké učení technické v Praze, Katedra fyzikální elektroniky [online]. 8. 3. 2000 [cit. 2022-06-09]. Dostupné z: http://kfe.fjfi.cvut.cz/~limpouch/numet/linalg/node24.html

[28] *Numerické řešení PDR*. Fakulta elektrotechniky a komunikačních technologií, VUT v Brně [online]. [2015] [cit. 2022-06-09]. Dostupné z: https://www.umat.fekt.vut.cz/~svobodaz/MKC-DRE/pocvi1.pdf

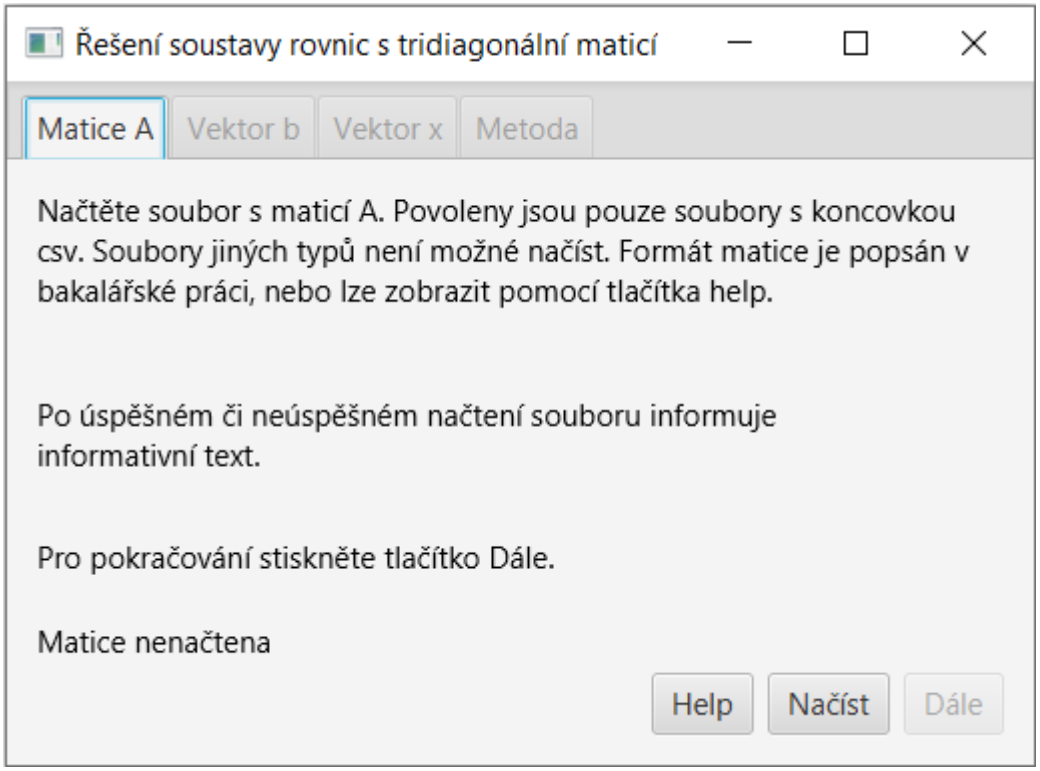
[29] TICHÝ, Petr. *Metoda sdružených gradientů*. Univerzita Karlova, Matematicko-fyzikální fakulta [online]. 4. 12. 2013 [cit. 2022-06-09]. Dostupné z: https://www2.karlin.mff.cuni.cz/~ptichy/blogs/nan/NA_09_prednaska.pdf

[30] HARAPES, Václav. *Numerické metody řešení soustav lineárních rovnic* [online]. Pardubice, 2011 [cit. 2022-06-09]. Dostupné z: https://dk.upce.cz/bitstream/handle/10195/42107/HarapesV_NumerickeMetody_JR_2011.pdf?sequence=4&isAllowed=y. Bakalářská práce. Univerzita Pardubice, Fakulta elektrotechniky a informatiky. Vedoucí práce RNDr. Josef Rak, Ph.D.

# PŘÍLOHY

# PŘÍLOHA A – UŽIVATELSKÁ PŘÍRUČKA

Na obrázku 5 je vidět grafické rozhraní aplikace.




Obrázek 5: Grafické rozhraní aplikace

Rozhraní aplikace je řešeno formou průvodce.

Po stisku tlačítka „Načíst“ se zobrazí dialogové okno pro výběr souboru s daty. Pokud jsou data úspěšně načteny, tak se například text „Matice nenačtena“ bude změněn na „Matice úspěšně načtena“ a uživatel může přejít na další kartu. Povelené formáty souborů jsou popsány přímo v aplikaci, kapitolách 4.4.1 a 4.4.2, nebo lze zobrazit správný formát dat po stisknutí tlačítka „Help“.

Tlačítka „Dále“ a „Zpět“ slouží pro přepínání mezi záložkami.

# PŘÍLOHA B – APLIKACE PRO ŘEŠENÍ SOUSTAV ROVNIC

V adresáři „Aplikace“ se nachází praktická část mé bakalářské práce – aplikace pro řešení soustav lineárních rovnic přibližnými metodami,

V adresáři „Zdrojový kód“ se nachází projekt pro vývojové studio NetBeans,

V adresáři „Příklad“ se nachází příklady matic pro výše uvedenou aplikaci.